Indian Institute of Science Education and Research, Mohali

Co-Starring Network Of Indian Actors

Athira Sreejith (MS18033)
Sourav S (MS18084)

Course	-	IDC407 - Network Science
Instructor	-	Dr Kuljeet Sandhu
Date of submission	-	24-04-2022

Contents

1 Introduction

Networks involve graphical representation of relationships (edges) between variables (nodes).Network analysis helps to estimate complex patterns of relationships and an analysis of network structures yields the core features of the network. They can also be used to investigate the social structures, and how these have evolved over the years. Here, we build a co-starring network of actors, and analyse the network attributes to infer if their dynamics are predictive of the broader political, socio-economic and technological changes.
Various studies have shown that the co-starring networks in Bollywood shows a clear connection to our political and economic history. These demonstrate how the industry has waxed and waned in response to the social, economic and political conditions of our country. In this project, we have presented an analysis of the co-starring networks of Indian movies, including regional film industries along with Bollywood. These are indicative of the culture and trends of not only the country as a whole, but also of each region that the industry is based on.
Although we have specifically focussed on co-starring actor network and its attribues in this project, the code has been written like a library with modules, and hence can be further used to study affiliation networks in general for any given dataset. The modules used in the main code have been discussed in subsequent sections.

```
#!/usr/bin/python3

from constructor import *
from visual import *
from analyse import *

import csv
import matplotlib.pyplot as plt
import numpy as np

#Loads the data set
with open('movies_data.csv', 'r', encoding='ISO-8859-1') as file:

        movie_dataset = list(csv.reader(file))


#Degree distribution

#----all-years

G = create_graph(movie_dataset)

degrees = [G.degree(i) for i in G.nodes]
deg_dist = np.unique(degrees, return_counts=True)
plt.plot(deg_dist[0], deg_dist[1], 'ro-')

plt.ylabel('frequency')
plt.xlabel('degree')
plt.grid(True)
plt.title('Degree distribution (1950-2022)')
plt.savefig('deg_dist_all_years.png')
#plt.show()
plt.clf()

#----year-wise

degs = interval_scan(movie_dataset, 'degrees', 10)

for i in range(len(degs)):
```

```
40
41      deg_dist = np.unique(degs[i], return_counts=True)
42
43      #to avoid log(0) since it is undefined, we remove elements with freq = 0
44      for j in range(len(deg_dist[1])):
45
46          if deg_dist[1][j] == 0:
47
48              del(deg_dist[1][j])
49              del(deg_dist[0][j])
50
51
52      plt.bar(np.log(deg_dist[0][1:]), np.log(deg_dist[1][1:]))
53
54      plt.ylabel('log(frequency)')
55      plt.xlabel('log(degree)')
56      plt.grid(True)
57      plt.title('Degree distribution log-log ({}-{})'.format(str(1950+10*i), str
        (1950+10*(i+1))))
58      plt.savefig('deg_dist_{}.png'.format(str(1950+10*i)))
59  # plt.show()
60      plt.clf()
61
62
63  #Network size
64
65  net_size = moving_scan(movie_dataset, 'nodes', 3)
66
67  plt.plot(range(1950,2021), net_size)
68  plt.ylabel('number of actors (3 year total)')
69  plt.xlabel('year')
70  plt.grid(True)
71  plt.title('Network Size')
72  plt.savefig('network_size.png')
73  #plt.show()
74  plt.clf()
75
76
77  #Network density
78
79  net_density = moving_scan(movie_dataset, 'density', 3)
80
81  plt.plot(range(1950,2021), net_density)
82  plt.ylabel('density (for 3 years)')
83  plt.xlabel('year')
84  plt.grid(True)
85  plt.title('Network Density')
86  plt.savefig('network_density.png')
87  #plt.show()
88  plt.clf()
89
90  #Centralities
91
92  deg_cen = interval_scan(movie_dataset, 'degree_centrality', 5)
93  btw_cen = interval_scan(movie_dataset, 'betweenness_centrality', 5)
94  cls_cen = interval_scan(movie_dataset, 'closeness_centrality', 5)
95  eig_cen = interval_scan(movie_dataset, 'eigenvector_centrality', 5)
96
97  dump_2_csv(top_n(deg_cen), 'deg_cen.csv')
98  dump_2_csv(top_n(btw_cen), 'btw_cen.csv')
```

```
dump_2_csv(top_n(cls_cen), 'cls_cen.csv')
dump_2_csv(top_n(eig_cen), 'eig_cen.csv')


#Visuals (pandemic and pre-pandemic)

#2018 and 2019
pre_pandemic = create_graph(movie_dataset, 2018, 2019)
display(pre_pandemic, 'pre_pandemic.html')

#2020 and 2021
pandemic = create_graph(movie_dataset, 2020, 2021)
display(pandemic, 'pandemic.html')
```

2 Data Collection

We collected data of Indian movies of different languages from 1950 to 2022 (`https://data.world/adrianmcmahon/imdb-dataset-all-indian-movies`). The collected data for each movie includes

- Name
- Lead actors
- Duration
- Year of release
- Genre

3 Constructing co-starring network

The Python package NetworkX was used to construct the co-starring network. We constructed a weighted network, with weight of each edge between two actors being the number of movies they have co-starred in.

3.1 Code

The commented code is given below.

```
#this module contains functions that help create a graph with a certain
    theme.

import networkx as nx
from itertools import combinations


#this function makes the nodes of the graph
def make_nodes(data, years, graph):

  actors = set()

  for i in data[1:]:

    if i[1] != '':

      if int(i[1][1:-1]) in years:
```

```
17
18          actors.update(i[-3:])
19
20    actors.remove('')
21
22    graph.add_nodes_from(actors, color = "#03DAC6")
23
24
25  #helps to make the graph weighted
26  def edge_weight(node_1, node_2, graph):
27
28    info =  graph.get_edge_data(node_1, node_2)
29
30    if info == None:
31
32      return 0
33
34    else:
35
36      return info['weight']
37
38
39  #this function makes the edgess of the graph
40  def make_edges(data, years, graph):
41
42    for i in data[1:]:
43
44      if i[1] != '':
45
46        #i[1] is basically the year in the csv data and [1:-1] removes
      unwanted start and end characters of the string
47        if int(i[1][1:-1]) in years:
48
49          affiliates = i[-3:]
50
51          if '' not in affiliates:
52
53            co_stars = list(combinations(affiliates, 2))
54
55           #weight is checked using edge_weight function and updated += 1
56            graph.add_edge(co_stars[0][0], co_stars[0][1], weight =
      edge_weight(co_stars[0][0], co_stars[0][1], graph) + 1, color = "#018786"
      )
57            graph.add_edge(co_stars[1][0], co_stars[1][1], weight =
      edge_weight(co_stars[1][0], co_stars[1][1], graph) + 1, color = "#018786"
      )
58
59          elif affiliates.count('') == 1:
60
61            affiliates.remove('')
62            graph.add_edge(affiliates[0], affiliates[1], weight = edge_weight(
      affiliates[0], affiliates[1], graph) + 1, color = "#018786")
63
64
65  #puts all the functions together to make the create_graph function which
      makes graph for a given time period
66  def create_graph(data, from_year=1950, till_year=2022):
67
68    G = nx.Graph()
69    years = list(range(from_year, till_year+1))
```

```
70
71      make_nodes(data, years, G)
72      make_edges(data, years, G)
73
74      return G
```

4 Visualisation

pyvis library (which uses HTML, CSS and JavaScript) was used for interactive visualisation.

4.1 Code

```
1  #a convenient function which takes in a networkx graph that can be called to
       make an interactive graph using pyvis.
2
3  from pyvis.network import Network
4
5  def display(graph, file_name='graph.html'):
6
7          nt = Network(height = "650px", width = "62%", bgcolor = "#222222",
       font_color = "white")
8          nt.from_nx(graph, show_edge_weights=False)
9          nt.barnes_hut()
10         nt.toggle_physics(True)
11         nt.show_buttons(filter_ = ["physics"])
12         nt.show(file_name)
13
14 #theme and layout is given coded here. Hence the user just needs to call the
       function with minimal params.
```

4.2 Results

Figure 1: Pre-pandemic period

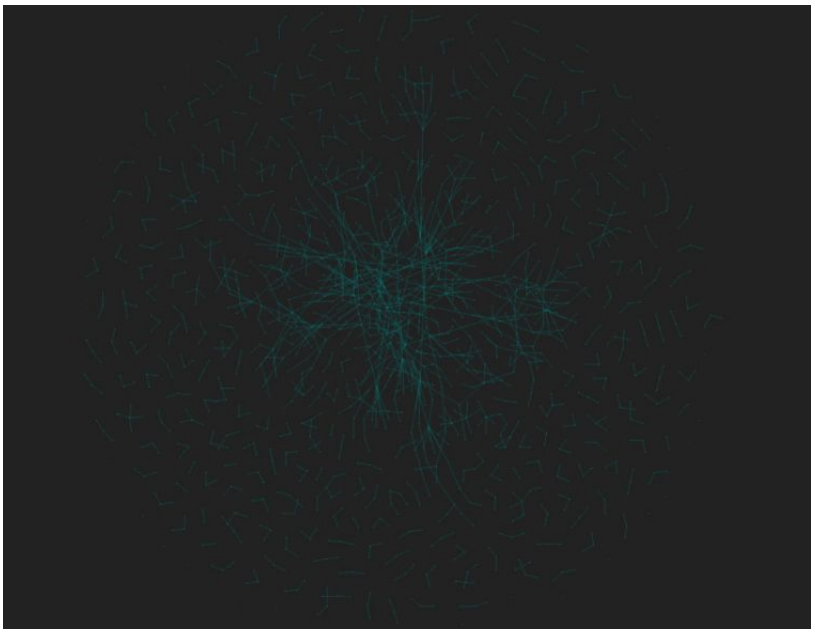


Figure 2: Pandemic period(with animation parameters)

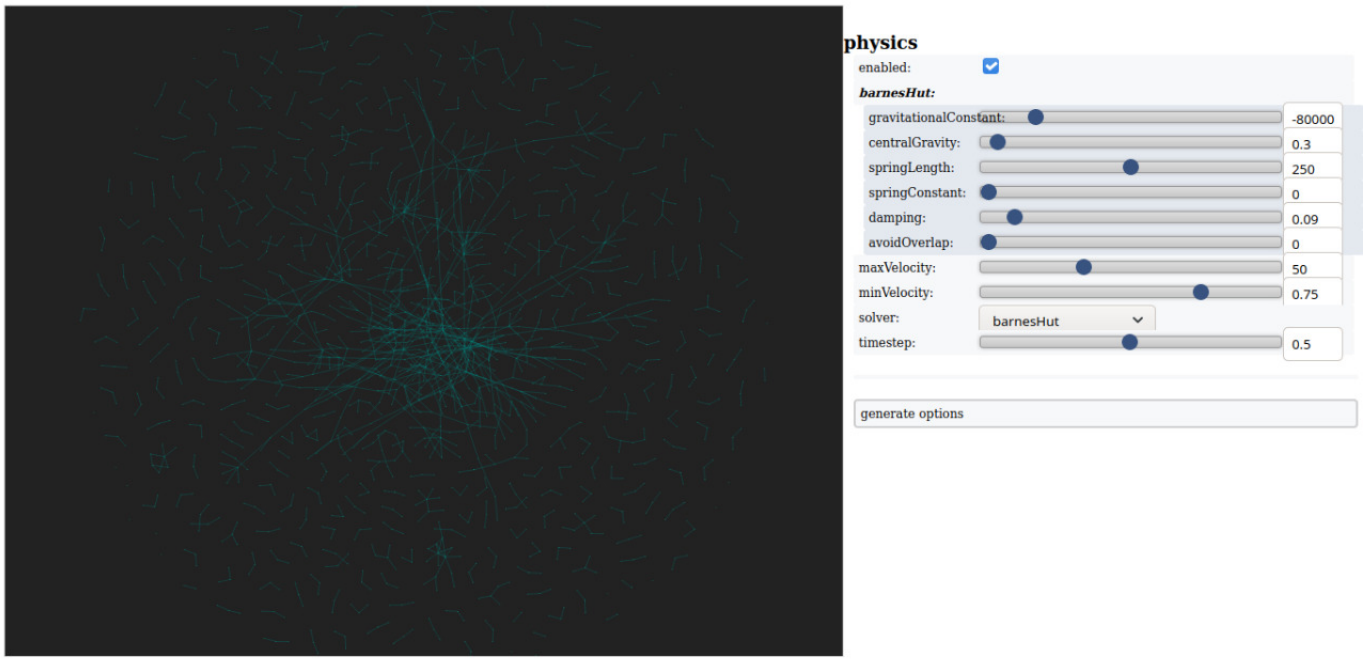


Figure 3: Pandemic period (zoomed in)

5 Analysis

The commented code is given below.

```
1 #this module has the main functions that will deal with calculating the
    attributes over time
```

```
2
3  from constructor import *
4  import numpy as np
5  import csv
6
7  #this function calculates the desired attribute by gradually moving through
       the years. Example: 1950-55, 1951-56, 1952-57 ...
8  def moving_scan(data, attribute, window, start=1950, end=2022):
9
10   y = []
11
12   for year in range(start, end-window+2):
13
14     G = create_graph(data, year, year+window-1)
15
16     print('calculating desired attribute: {} for year {} to {}'.format(
       attribute, year, year + window))
17
18     if attribute == 'nodes':
19
20       y.append(G.number_of_nodes())
21
22     elif attribute == 'degrees':
23
24       degrees = [G.degree(i) for i in G.nodes]
25       y.append(np.unique(degrees, return_counts=True))
26
27     elif attribute == 'degree_centrality':
28
29       y.append(nx.degree_centrality(G))
30
31     elif attribute == 'closeness_centrality':
32
33       y.append(nx.closeness_centrality(G))
34
35     elif attribute == 'betweenness_centrality':
36
37       y.append(nx.betweenness_centrality(G))
38
39     elif attribute == 'eigenvector_centrality':
40
41       y.append(nx.eigenvector_centrality(G))
42
43     elif attribute == 'density':
44
45       m = G.number_of_edges()
46       n = G.number_of_nodes()
47
48       y.append(2*m/(n*(n-1)))
49
50   return y
51
52 #this function calculates the desired attribute by moving through discrete
       class intervals. Example: 1950-55, 1955-60, 1965-70 ...
53 def interval_scan(data, attribute, window, start=1950, end=2022):
54
55   y = []
56   n = 0
57
58   year = start + n * window
```

```
59
60
61      while year < end:
62
63          G = create_graph(data, year, year + window)
64
65          print('calculating desired attribute: {} for year {} to {}'.format(
            attribute, year, year + window))
66
67          if attribute == 'nodes':
68
69              y.append(G.number_of_nodes())
70
71          elif attribute == 'degrees':
72
73              degrees = [G.degree(i) for i in G.nodes]
74              y.append(np.unique(degrees, return_counts=True))
75
76          elif attribute == 'degree_centrality':
77
78              y.append(nx.degree_centrality(G))
79
80          elif attribute == 'closeness_centrality':
81
82              y.append(nx.closeness_centrality(G))
83
84          elif attribute == 'betweenness_centrality':
85
86              y.append(nx.betweenness_centrality(G))
87
88          elif attribute == 'eigenvector_centrality':
89
90              y.append(nx.eigenvector_centrality(G))
91
92          elif attribute == 'density':
93
94              m = G.number_of_edges()
95              n = G.number_of_nodes()
96
97              y.append(2*m/(n*(n-1)))
98
99          n += 1
100         year = start + n * window
101
102     return y
103
104
105 #returns list of top n (for desired attribute) for every year
106 def top_n(_list, top=10):
107
108     m = []
109
110     print('Making a list of top {} for every year...'.format(str(top)))
111
112     for i in _list:
113
114         top_list = sorted([(v, k) for k, v in i.items()], reverse=True)[:top]
115         m.append([(round(i[0],4), i[1]) for i in top_list])
116
117     return m
```

```
118
119
120 #dumps the top_n out list to csv file
121 def dump_2_csv(_list, file):
122
123   with open(file, 'w') as f:
124
125     write = csv.writer(f)
126
127     for i in _list:
128       for j in i:
129
130         write.writerow([j[1], j[0]])
131
132       write.writerow(['', ''])
133
134   print('Successfully dumped to {}'.format(file))
```

```
calculating desired attribute: degree_centrality for year 2010 to 2015
calculating desired attribute: degree_centrality for year 2015 to 2020
calculating desired attribute: degree_centrality for year 2020 to 2025
calculating desired attribute: closeness_centrality for year 1950 to 1955
calculating desired attribute: closeness_centrality for year 1955 to 1960
calculating desired attribute: closeness_centrality for year 1960 to 1965
calculating desired attribute: closeness_centrality for year 1965 to 1970
calculating desired attribute: closeness_centrality for year 1970 to 1975
calculating desired attribute: closeness_centrality for year 1975 to 1980
calculating desired attribute: closeness_centrality for year 1980 to 1985
calculating desired attribute: closeness_centrality for year 1985 to 1990
calculating desired attribute: closeness_centrality for year 1990 to 1995
calculating desired attribute: closeness_centrality for year 1995 to 2000
calculating desired attribute: closeness_centrality for year 2000 to 2005
calculating desired attribute: closeness_centrality for year 2005 to 2010
calculating desired attribute: closeness_centrality for year 2010 to 2015
calculating desired attribute: closeness_centrality for year 2015 to 2020
calculating desired attribute: closeness_centrality for year 2020 to 2025
calculating desired attribute: eigenvector_centrality for year 1950 to 1955
calculating desired attribute: eigenvector_centrality for year 1955 to 1960
calculating desired attribute: eigenvector_centrality for year 1960 to 1965
calculating desired attribute: eigenvector_centrality for year 1965 to 1970
calculating desired attribute: eigenvector_centrality for year 1970 to 1975
calculating desired attribute: eigenvector_centrality for year 1975 to 1980
calculating desired attribute: eigenvector_centrality for year 1980 to 1985
calculating desired attribute: eigenvector_centrality for year 1985 to 1990
calculating desired attribute: eigenvector_centrality for year 1990 to 1995
calculating desired attribute: eigenvector_centrality for year 1995 to 2000
calculating desired attribute: eigenvector_centrality for year 2000 to 2005
calculating desired attribute: eigenvector_centrality for year 2005 to 2010
calculating desired attribute: eigenvector_centrality for year 2010 to 2015
calculating desired attribute: eigenvector_centrality for year 2015 to 2020
calculating desired attribute: eigenvector_centrality for year 2020 to 2025
Making a list of top 10 for every year...
Successfully dumped to deg_cen.csv
Making a list of top 10 for every year...
Successfully dumped to cls_cen.csv
Making a list of top 10 for every year...
Successfully dumped to eig_cen.csv
```

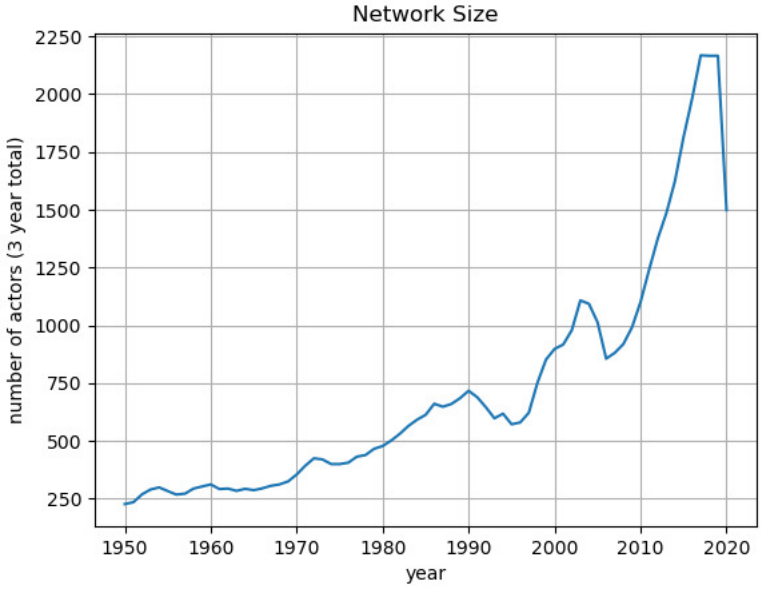
Figure 4: Output of the code

5.1 Evolution of network size

The fluctuations in the network size can be attributed to the socio-economic and political changes. The variation in the network size over time was studied by plotting no.of nodes per year, as shown below.

From the plot, it can be clearly seen that the network size has increased over the years. This general trend is related to the expansion of the film industry and the increasing demand.
A more interesting feature in these plots are the variations in the rate of increase of the network size. A close examination of the rate of change of network size over the years shows a slow increase till 1990s, which can be attributed to the socio-economic growth the country witnessed during that

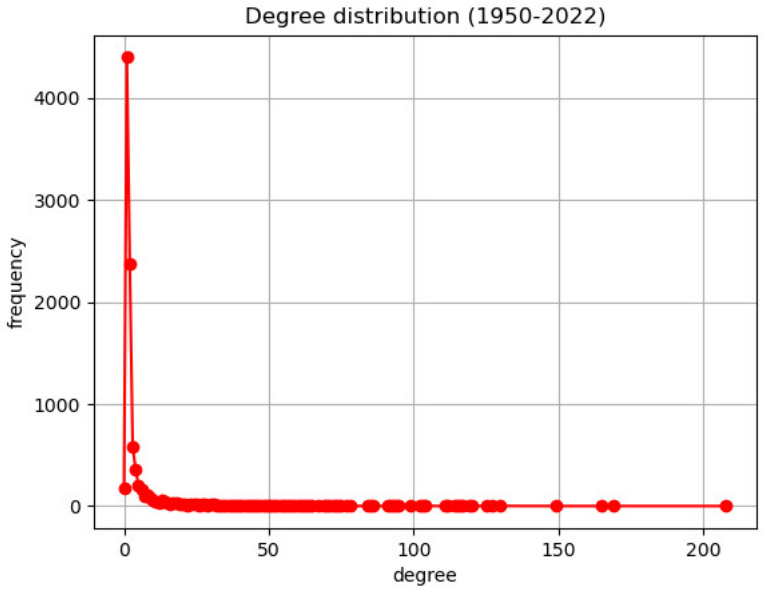
Figure 5: Pandemic period (zoomed in)

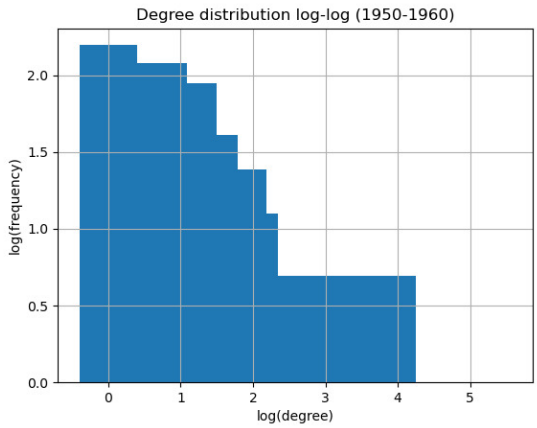


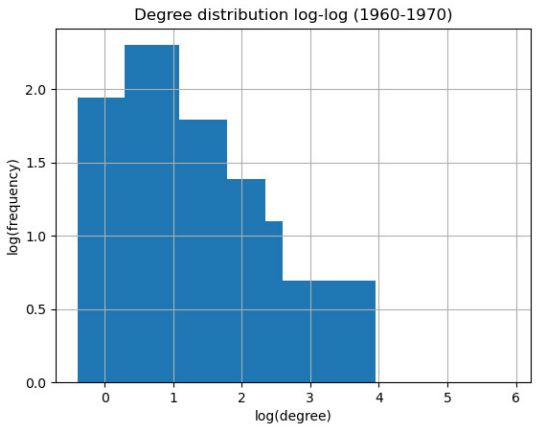
period. The rate of increase is more rapid from about 1995-2005, and from 2005-2019. This is possibly due to economic liberation.
We find a sudden decrease in the network size during the years 1990-1995. During this period, there was increasing violence, declining musical quality and rise in video piracy. This contributed to the decline in the network size. The decline in network size at 2020 is primarily due to the pandemic.

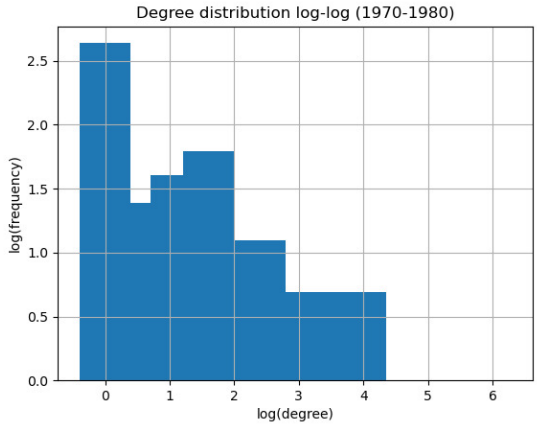
5.2 Degree distribution

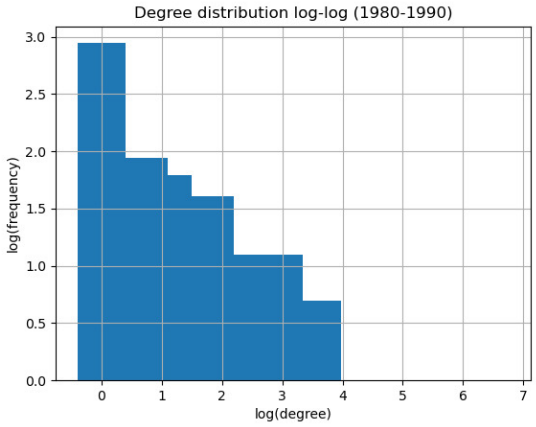
Degree distribution of networks provides insight into some of the structural features such as the existence of hubs in the network. Most real world networks exhibit right skewed degree distributions, with a large of nodes having small degrees and a small number of nodes having large degrees. The features of degree distribution of the network over the years were studied as shown below.

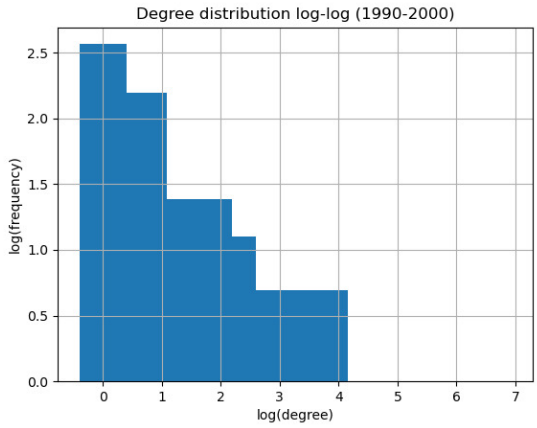


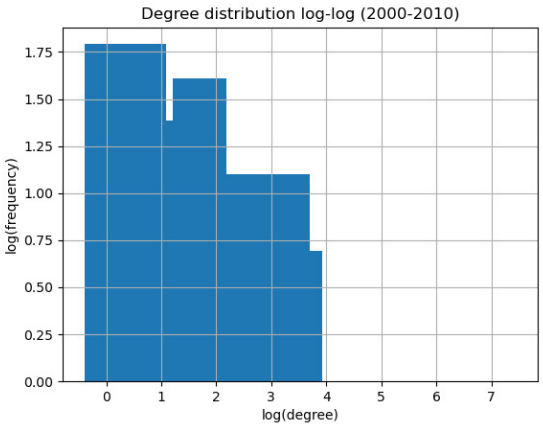

Degree distribution log-log (1950-1960)
log(frequency)
log(degree)

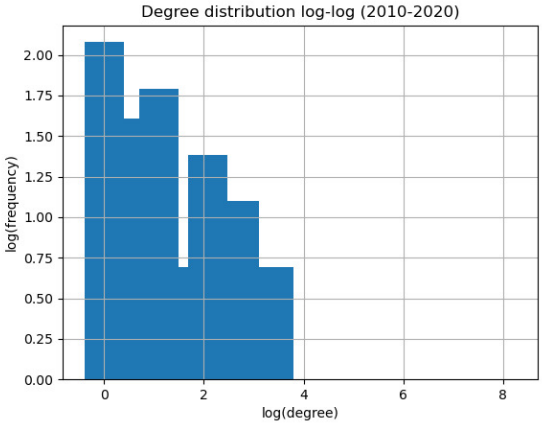

Degree distribution log-log (1960-1970)
log(frequency)
log(degree)

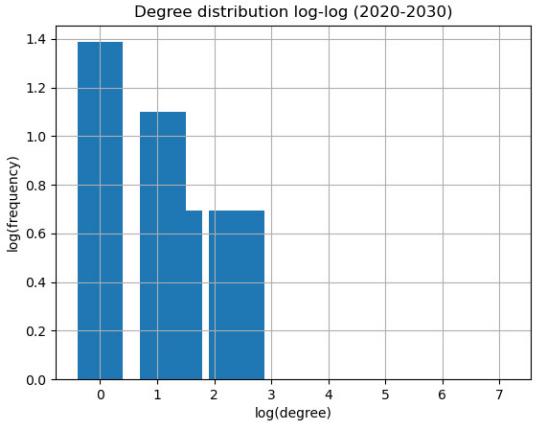

Degree distribution log-log (1970-1980)
log(frequency)
log(degree)


Degree distribution log-log (1980-1990)
log(frequency)
log(degree)


Degree distribution log-log (1990-2000)
log(frequency)
log(degree)


Degree distribution log-log (2000-2010)
log(frequency)
log(degree)

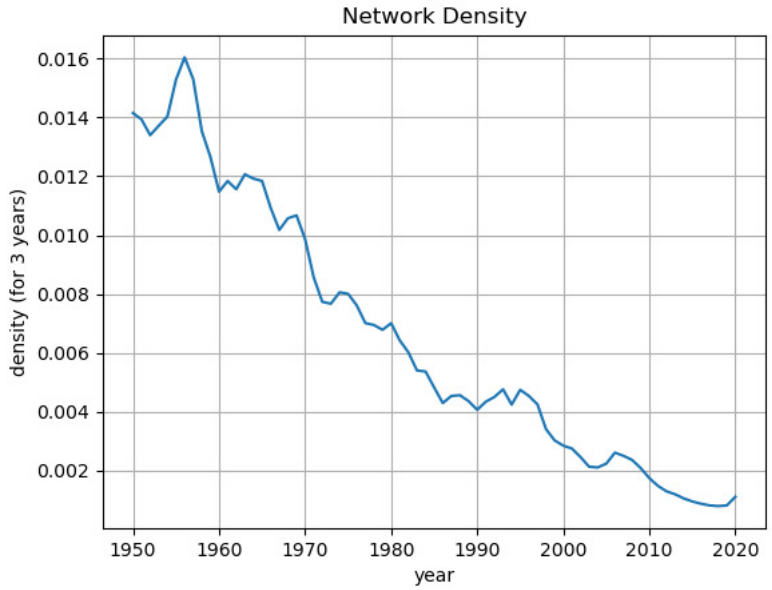

Degree distribution log-log (2010-2020)
log(frequency)
log(degree)


Degree distribution log-log (2020-2030)
log(frequency)
log(degree)

The first plot is right-skewed as is expected in most real world networks. A log-log plot of the normalised degree shows a linear trend in all the time intervals considered. This indicates that the power law relation of the network holds throughout the years.

5.3 Network density

Network density or the proportion of direct ties in a network relative to the total number possible, is a measure of the prevalence of direct tie within a social network. This feature of the co-starring network and its evolution over the years have been looked into.



From the plot, we can see that the network density has decreased over the years. This is possibly due to the rise in the number of nodes, while the number of collaborations didn't increase proportionately.

5.4 Centrality Measures

We looked into 4 centrality measures of the network during time intervals of 10 years. The centrality measures used are

- Degree Centrality
- Closeness Centrality
- Betweenness Centrality
- Eigenvector Centrality

The centrality measures of the top 10 nodes for each time period is given below.

Table 1: 1950-55

Degree Centrality		Closeness Centrality		Betweenness Centrality		Eigenvector Centrality	
Geeta Bali	0.1082	Geeta Bali	0.374	Geeta Bali	0.0971	Geeta Bali	0.2616
Dev Anand	0.0871	Agha	0.3703	Agha	0.0795	Agha	0.2363
Bharat Bhushan	0.0844	Jeevan	0.3619	Dev Anand	0.0675	Jeevan	0.2193
Agha	0.0844	Shyama	0.3559	Bharat Bhushan	0.0667	Dev Anand	0.2103
Jeevan	0.0739	Dev Anand	0.3538	Shyama	0.0565	Bharat Bhushan	0.2045
Ashok Kumar	0.0739	Bharat Bhushan	0.3534	Master Bhagwan	0.056	Karan Dewan	0.1981
Karan Dewan	0.0712	Suraiya	0.3517	Jeevan	0.0534	Ashok Kumar	0.1918
Shyama	0.0686	Karan Dewan	0.3457	Ashok Kumar	0.0515	Suraiya	0.1845
Madhubala	0.0686	Pran	0.3418	David Abraham	0.0459	Shyama	0.1839
Suraiya	0.0607	Madhubala	0.3418	Madhubala	0.0444	Madhubala	0.1703

Table 2: 1955-60

Degree Centrality		Closeness Centrality		Betweenness Centrality		Eigenvector Centrality	
Pradeep Kumar	0.0975	Ajit	0.3798	Mahipal	0.0726	Pradeep Kumar	0.2352
Ajit	0.085	Pradeep Kumar	0.3793	Ajit	0.0678	Shyama	0.2016
Ashok Kumar	0.0825	Shyama	0.3752	Dev Anand	0.0555	Ashok Kumar	0.1897
Kishore Kumar	0.08	Helen	0.3732	Chitra	0.0527	Kishore Kumar	0.1854
Shyama	0.0775	Mahipal	0.3728	Kamran	0.052	Ajit	0.1822
Mahipal	0.0775	Nirupa Roy	0.3696	Pradeep Kumar	0.0511	Johnny Walker	0.1703
Dev Anand	0.0775	Ashok Kumar	0.3657	Nirupa Roy	0.0459	Helen	0.1697
Nirupa Roy	0.075	Chitra	0.3646	Master Bhagwan	0.0456	Mahipal	0.1663
Chitra	0.075	Master Bhagwan	0.3642	Ashok Kumar	0.044	Meena Kumari	0.1572
Helen	0.0725	Geeta Bali	0.3642	Habib	0.0423	Chitra	0.1533

Table 3: 1960-65

Degree Centrality		Closeness Centrality		Betweenness Centrality		Eigenvector Centrality	
Helen	0.086	Helen	0.3535	Chitra	0.0817	Ashok Kumar	0.2733
Ashok Kumar	0.0791	Ashok Kumar	0.3378	Helen	0.0799	Helen	0.2368
Chitra	0.0698	Master Bhagwan	0.3344	Mahipal	0.0588	Mehmood	0.1884
Dara Singh	0.0651	Mehmood	0.3324	Mehmood	0.0575	Mala Sinha	0.1758
Mehmood	0.0605	Chitra	0.3285	Ashok Kumar	0.0574	Rajendra Kumar	0.1555
Paidi Jairaj	0.0581	Dara Singh	0.3234	Paidi Jairaj	0.0532	Sunil Dutt	0.1538
Kishore Kumar	0.0581	Ajit	0.3234	Dara Singh	0.05	Master Bhagwan	0.1535
Dev Anand	0.0581	Vijaya Choudhury	0.3215	Master Bhagwan	0.0474	Ajit	0.1517
Pradeep Kumar	0.0558	Paidi Jairaj	0.3194	Manher Desai	0.0433	Dara Singh	0.1466
Shammi Kapoor	0.0535	Jeevan	0.3194	Padmini	0.042	Pradeep Kumar	0.1461

Table 4: 1965-70

Degree Centrality		Closeness Centrality		Betweenness Centrality		Eigenvector Centrality	
Ashok Kumar	0.0887	Helen	0.3253	Ashok Kumar	0.1045	Ashok Kumar	0.2666
Sanjeev Kumar	0.082	Ashok Kumar	0.3243	Sanjeev Kumar	0.0897	Dharmendra	0.2106
Dara Singh	0.071	Sanjeev Kumar	0.3201	Helen	0.0761	Sanjeev Kumar	0.2034
Helen	0.0665	Mumtaz	0.3157	Dara Singh	0.0747	Mumtaz	0.1939
Mumtaz	0.0643	Dara Singh	0.3054	Feroz Khan	0.0535	Asha Parekh	0.172
Feroz Khan	0.0599	Feroz Khan	0.3042	Mumtaz	0.0529	Sunil Dutt	0.1717
Dharmendra	0.0599	Balraj Sahni	0.2996	Sheikh Mukhtar	0.0383	Pran	0.1698
Master Bhagwan	0.051	Prithviraj Kapoor	0.2971	Prithviraj Kapoor	0.0339	Balraj Sahni	0.1693
Biswajit Chatterjee	0.051	Sunil Dutt	0.2963	Balraj Sahni	0.0332	Feroz Khan	0.1652
Sheikh Mukhtar	0.0488	Sheikh Mukhtar	0.2963	Master Bhagwan	0.033	Biswajit Chatterjee	0.1585

Table 5: 1970-75

Degree Centrality		Closeness Centrality		Betweenness Centrality		Eigenvector Centrality	
Sanjeev Kumar	0.0823	Sanjeev Kumar	0.3054	Sanjeev Kumar	0.0857	Ashok Kumar	0.2383
Rajesh Khanna	0.0652	Shatrughan Sinha	0.3011	Hema Malini	0.0535	Rajesh Khanna	0.2364
Ashok Kumar	0.06	Ashok Kumar	0.2981	N.T. Rama Rao	0.0447	Sanjeev Kumar	0.2342
Dharmendra	0.0583	Dharmendra	0.293	Sanjay Khan	0.0389	Dharmendra	0.2333
Vinod Khanna	0.0549	Sanjay Khan	0.292	Jeevan	0.0387	Shatrughan Sinha	0.2123
Shatrughan Sinha	0.0532	Mumtaz	0.2918	Shashi Kapoor	0.0386	Vinod Khanna	0.2059
Shashi Kapoor	0.0515	Amitabh Bachchan	0.2916	Navin Nischol	0.0376	Jeetendra	0.1816
Sanjay Khan	0.0463	Rajesh Khanna	0.2892	Mumtaz	0.0373	Amitabh Bachchan	0.1737
Navin Nischol	0.0446	Yogeeta Bali	0.2872	Rakesh Pandey	0.0364	Mumtaz	0.1699
Mumtaz	0.0446	Vinod Khanna	0.2863	Rajesh Khanna	0.0357	Hema Malini	0.1697

Table 6: 1975-80

Degree Centrality		Closeness Centrality		Betweenness Centrality		Eigenvector Centrality	
Sanjeev Kumar	0.0668	Shashi Kapoor	0.2734	Sanjeev Kumar	0.072	Shashi Kapoor	0.2804
Shashi Kapoor	0.0637	Sanjeev Kumar	0.2727	Helen	0.0648	Ashok Kumar	0.2462
Ashok Kumar	0.059	Dharmendra	0.2689	Jagdeep	0.056	Sanjeev Kumar	0.2432
Vinod Khanna	0.0575	Ashok Kumar	0.2684	Rakesh Pandey	0.0525	Vinod Khanna	0.2265
Rajesh Khanna	0.0543	Bindu	0.2667	Jeetendra	0.0512	Jeetendra	0.2044
Jeetendra	0.0543	Jeetendra	0.266	Bindu	0.0511	Shatrughan Sinha	0.204
Shatrughan Sinha	0.0528	Vinod Khanna	0.2655	N.T. Rama Rao	0.048	Rajesh Khanna	0.2037
Vinod Mehra	0.0481	Yogeeta Bali	0.2609	Shashi Kapoor	0.0467	Sunil Dutt	0.1933
Sunil Dutt	0.0481	Asrani	0.2604	Dharmendra	0.0445	Dharmendra	0.1825
Dharmendra	0.0466	Shatrughan Sinha	0.2589	Rajinikanth	0.0426	Rekha	0.178

Table 7: 1980-85

Degree Centrality		Closeness Centrality		Betweenness Centrality		Eigenvector Centrality	
Mithun Chakraborty	0.0713	Mithun Chakraborty	0.2915	Mithun Chakraborty	0.0773	Jeetendra	0.2501
Raj Babbar	0.0639	Raj Babbar	0.2838	Raj Babbar	0.0585	Raj Babbar	0.2404
Jeetendra	0.0565	Jeetendra	0.2775	Bharat Bhushan	0.0493	Mithun Chakraborty	0.23
Rajesh Khanna	0.0504	Vinod Mehra	0.2773	Master Bhagwan	0.049	Shashi Kapoor	0.23
Shashi Kapoor	0.0455	Shashi Kapoor	0.2752	Vinod Mehra	0.0454	Rajesh Khanna	0.2174
Sanjeev Kumar	0.0442	Amjad Khan	0.2741	Kamal Haasan	0.0441	Sanjeev Kumar	0.2093
Kamal Haasan	0.043	Sanjeev Kumar	0.2714	Rajinikanth	0.0428	Dharmendra	0.2027
Amitabh Bachchan	0.0418	Amitabh Bachchan	0.2714	Shashi Kapoor	0.04	Shatrughan Sinha	0.1906
Vinod Mehra	0.0405	Dharmendra	0.2711	Rati Agnihotri	0.0367	Amitabh Bachchan	0.1814
Dharmendra	0.0405	Ashok Kumar	0.271	Asrani	0.0365	Vinod Mehra	0.1718

Table 8: 1985-90

Degree Centrality		Closeness Centrality		Betweenness Centrality		Eigenvector Centrality	
Mithun Chakraborty	0.0675	Mithun Chakraborty	0.2924	Mithun Chakraborty	0.1074	Mithun Chakraborty	0.296
Raj Babbar	0.0487	Raj Babbar	0.2668	Sadashiv Amrapurkar	0.0737	Raj Babbar	0.2227
Rajesh Khanna	0.0417	Rajesh Khanna	0.2648	Raj Babbar	0.0433	Dharmendra	0.2225
Dharmendra	0.0417	Govinda	0.2646	Rajesh Khanna	0.0416	Jeetendra	0.2154
Jeetendra	0.0407	Sadashiv Amrapurkar	0.2643	Govinda	0.036	Govinda	0.2015
Govinda	0.0377	Dharmendra	0.2642	Hemant Birje	0.0354	Jackie Shroff	0.1905
Jackie Shroff	0.0357	Raj Kiran	0.263	Raj Kiran	0.0342	Rajesh Khanna	0.1861
Shatrughan Sinha	0.0318	Rekha	0.2626	Dharmendra	0.032	Rishi Kapoor	0.1777
Anil Kapoor	0.0318	Jeetendra	0.2618	Rajinikanth	0.0316	Shatrughan Sinha	0.177
Sadashiv Amrapurkar	0.0288	Jackie Shroff	0.2618	Anuradha	0.0301	Sanjay Dutt	0.1712

Table 9: 1990-95

Degree Centrality		Closeness Centrality		Betweenness Centrality		Eigenvector Centrality	
Mithun Chakraborty	0.0482	Mithun Chakraborty	0.2676	Mithun Chakraborty	0.0896	Jeetendra	0.2611
Jeetendra	0.0403	Dharmendra	0.2575	Sadashiv Amrapurkar	0.0623	Dharmendra	0.2481
Dharmendra	0.0403	Sadashiv Amrapurkar	0.2536	Rakesh Bedi	0.0508	Mithun Chakraborty	0.2434
Sadashiv Amrapurkar	0.0364	Jeetendra	0.2535	Rita Bhaduri	0.0446	Jackie Shroff	0.2019
Govinda	0.0344	Govinda	0.2513	Disco Shanti	0.0414	Govinda	0.2014
Sanjay Dutt	0.0334	Juhi Chawla	0.2473	Jeetendra	0.0387	Vinod Khanna	0.1986
Raj Babbar	0.0334	Rita Bhaduri	0.2442	Dharmendra	0.0384	Rishi Kapoor	0.1917
Rakesh Bedi	0.0324	Ishrat Ali	0.2438	Juhi Chawla	0.0366	Sanjay Dutt	0.1852
Rishi Kapoor	0.0315	Raj Babbar	0.2429	Laxmikant Berde	0.0344	Aditya Pancholi	0.1704
Jackie Shroff	0.0315	Asrani	0.2429	Akshay Kumar	0.0326	Madhuri Dixit	0.1549

Table 10: 1995-2000

Degree Centrality		Closeness Centrality		Betweenness Centrality		Eigenvector Centrality	
Mithun Chakraborty	0.0588	Mithun Chakraborty	0.2472	Mithun Chakraborty	0.1239	Mithun Chakraborty	0.2459
Shakti Kapoor	0.0386	Dharmendra	0.2408	Shakti Kapoor	0.0763	Govinda	0.2107
Govinda	0.0312	Govinda	0.2305	Dharmendra	0.0651	Sunil Shetty	0.2101
Dharmendra	0.0312	Shakti Kapoor	0.2289	Govinda	0.0457	Akshay Kumar	0.2046
Sunil Shetty	0.0294	Ishrat Ali	0.2283	Asrani	0.0431	Jackie Shroff	0.1998
Jackie Shroff	0.0294	Asrani	0.2264	Akshay Kumar	0.0342	Sanjay Dutt	0.1776
Akshay Kumar	0.0275	Akshay Kumar	0.2246	Ishrat Ali	0.0327	Karisma Kapoor	0.1753
Ishrat Ali	0.0266	Aditya Pancholi	0.2221	Sadashiv Amrapurkar	0.0269	Anil Kapoor	0.172
Sanjay Dutt	0.0257	Jackie Shroff	0.2209	Sanjay Dutt	0.0261	Dharmendra	0.1658
Sadashiv Amrapurkar	0.0239	Ajay Devgn	0.2196	Sunny Deol	0.0234	Ajay Devgn	0.1658

Table 11: 2000-05

Degree Centrality		Closeness Centrality		Betweenness Centrality		Eigenvector Centrality	
Amitabh Bachchan	0.0243	Amitabh Bachchan	0.2137	Mohan Joshi	0.062	Amitabh Bachchan	0.3053
Shakti Kapoor	0.0237	Mohan Joshi	0.2102	Amitabh Bachchan	0.06	Ajay Devgn	0.2554
Ajay Devgn	0.0206	Dharmendra	0.2079	Shakti Kapoor	0.0529	Sanjay Dutt	0.232
Sunil Shetty	0.02	Ajay Devgn	0.2074	Dharmendra	0.0489	Sunil Shetty	0.2287
Sanjay Dutt	0.0182	Sunil Shetty	0.207	Tabu	0.0434	Akshay Kumar	0.2159
Mohan Joshi	0.0182	Sanjay Dutt	0.204	Raza Murad	0.0384	Salman Khan	0.1985
Amit Pachori	0.0176	Shakti Kapoor	0.2034	Sunil Shetty	0.0372	Abhishek Bachchan	0.1841
Satnam Kaur	0.017	Govinda	0.2004	Laxmikant Berde	0.0354	Jackie Shroff	0.1794
Sapna Sappu	0.017	Ishrat Ali	0.2	Govinda	0.0353	Anil Kapoor	0.1791
Raza Murad	0.017	Jackie Shroff	0.1984	Mithun Chakraborty	0.0332	Rani Mukerji	0.157

Table 12: 2005-10

Degree Centrality		Closeness Centrality		Betweenness Centrality		Eigenvector Centrality	
Amitabh Bachchan	0.0254	Ajay Devgn	0.1871	Anupam Kher	0.0525	Akshay Kumar	0.3276
Akshay Kumar	0.0229	Paresh Rawal	0.1834	Shakti Kapoor	0.0478	Ajay Devgn	0.3153
Ajay Devgn	0.0211	Anupam Kher	0.1827	Amitabh Bachchan	0.0411	Amitabh Bachchan	0.268
Salman Khan	0.018	Akshay Kumar	0.1814	Milind Gunaji	0.0386	Sanjay Dutt	0.2573
Sanjay Dutt	0.0174	Amitabh Bachchan	0.1802	Ajay Devgn	0.0337	Salman Khan	0.1963
Mithun Chakraborty	0.0149	Sanjay Dutt	0.1801	Rati Agnihotri	0.0319	Priyanka Chopra Jonas	0.194
Anupam Kher	0.0149	Salman Khan	0.178	Mithun Chakraborty	0.0305	John Abraham	0.1936
John Abraham	0.0136	Priyanka Chopra Jonas	0.1771	Mahima Chaudhry	0.0292	Sunil Shetty	0.1672
Priyanka Chopra Jonas	0.013	Kay Kay Menon	0.1741	Akshay Kumar	0.0288	Paresh Rawal	0.1617
Kay Kay Menon	0.013	Om Puri	0.1737	Paresh Rawal	0.0273	Saif Ali Khan	0.1495

Table 13: 2010-15

Degree Centrality		Closeness Centrality		Betweenness Centrality		Eigenvector Centrality	
Akshay Kumar	0.0156	Akshay Kumar	0.1338	Akshay Kumar	0.0287	Akshay Kumar	0.3241
Anupam Kher	0.0112	Anupam Kher	0.1324	Anupam Kher	0.0258	Ajay Devgn	0.2375
Ajay Devgn	0.0112	Paresh Rawal	0.1306	Asrani	0.0237	Anupam Kher	0.2109
Emraan Hashmi	0.0103	Emraan Hashmi	0.1302	Rajpal Yadav	0.0225	Kareena Kapoor	0.2095
Asrani	0.0089	Naseeruddin Shah	0.1296	Nawazuddin Siddiqui	0.0207	Amitabh Bachchan	0.1794
Amitabh Bachchan	0.0089	Amitabh Bachchan	0.1286	Manoj Bajpayee	0.0192	Sanjay Dutt	0.1752
Sanjay Dutt	0.0085	Kareena Kapoor	0.1283	Om Puri	0.0166	Deepika Padukone	0.1668
Nawazuddin Siddiqui	0.0085	Ajay Devgn	0.128	Naseeruddin Shah	0.016	Riteish Deshmukh	0.1539
Manoj Bajpayee	0.0085	Riteish Deshmukh	0.128	Emraan Hashmi	0.0159	Salman Khan	0.1523
Shahid Kapoor	0.008	Sanjay Dutt	0.1279	Jimmy Sheirgill	0.0152	Priyanka Chopra Jonas	0.1505

Table 14: 2015-20

Degree Centrality		Closeness Centrality		Betweenness Centrality		Eigenvector Centrality	
Akshay Kumar	0.0079	Nawazuddin Siddiqui	0.099	Sanjay Mishra	0.0212	Manoj Bajpayee	0.3186
Nawazuddin Siddiqui	0.0076	Manoj Bajpayee	0.0987	Nawazuddin Siddiqui	0.0201	Akshay Kumar	0.2675
Manoj Bajpayee	0.0076	Ayushmann Khurrana	0.0972	Hrishitaa Bhatt	0.0167	Amitabh Bachchan	0.2363
Muntazir Ahmad	0.0074	Richa Chadha	0.0965	Richa Chadha	0.0148	Taapsee Pannu	0.2125
Sanjay Mishra	0.0068	Amitabh Bachchan	0.0962	Manoj Bajpayee	0.0138	Rajkummar Rao	0.177
Jimmy Sheirgill	0.0065	Sanjay Mishra	0.0956	Rajkummar Rao	0.013	Nawazuddin Siddiqui	0.1698
Rajkummar Rao	0.0062	Radhika Apte	0.0949	Pankaj Tripathi	0.0122	Ayushmann Khurrana	0.1641
Amitabh Bachchan	0.0062	Rajkummar Rao	0.0945	Dharmendra	0.0115	Jimmy Sheirgill	0.1637
Adil Hussain	0.0054	Akshay Kumar	0.0934	Muntazir Ahmad	0.0112	John Abraham	0.1458
Vinay Pathak	0.0051	Vijay Raaz	0.0933	Ayushmann Khurrana	0.0109	Varun Dhawan	0.145

Table 15: 2020-22

Degree Centrality		Closeness Centrality		Betweenness Centrality		Eigenvector Centrality	
Amitabh Bachchan	0.01	Ajay Devgn	0.0659	Vikrant Massey	0.0247	Amitabh Bachchan	0.4837
Sanjay Mishra	0.0093	Deepika Padukone	0.0652	Deepika Padukone	0.022	Ajay Devgn	0.3651
Nawazuddin Siddiqui	0.008	Saif Ali Khan	0.0652	Ajay Devgn	0.0208	Rakul Preet Singh	0.3351
Sharib Hashmi	0.0067	Alia Bhatt	0.064	Nawazuddin Siddiqui	0.0182	Akshay Kumar	0.2074
Rajpal Yadav	0.0067	Ranbir Kapoor	0.0635	Sharib Hashmi	0.0175	Neena Gupta	0.1881
Akshay Kumar	0.0067	Vikrant Massey	0.063	Ranbir Kapoor	0.0166	Jacqueline Fernandez	0.1871
Ajay Devgn	0.0067	Sharad Kelkar	0.0625	Sanjay Mishra	0.0162	Sidharth Malhotra	0.1828
Vikrant Massey	0.006	Jacqueline Fernandez	0.0625	Sharad Kelkar	0.0151	Vijay Raaz	0.135
Manoj Bakshi	0.006	Akshay Kumar	0.0624	Pankaj Tripathi	0.0143	Alia Bhatt	0.1338
Deepika Padukone	0.006	Rakul Preet Singh	0.0623	Richa Chadha	0.013	Ayushmann Khurrana	0.1329

6 Conclusion

The co-starring network for Indian films from 1950 was constructed and analysed. We observed correlations between the evolution of the network attributes and the socio-economic, cultural and technological changes.

7 Short-comings and future scope

- The visualisations can be further enhanced by adding centrality dependent colours for the nodes
- The data we obtained has a greater contribution from Bollywood movies, and this results in a bias in the analysis.
- We can create clusters in the network corresponding to different languages, as movies based on the same language are expected form a cluster. The analysis carried out globally can be extended locally to these language clusters.

References

https://www.technologyreview.com/2014/06/25/172208/how-the-bollywood-actors-network-reve
https://arxiv.org/pdf/1406.4607.pdf
https://arxiv.org/pdf/1407.3345.pdf
https://bookdown.org/omarlizardo/_main/2-9-density.html